Software agents in molecular computational biology

John W. Keele and James E. Wray

Date received (in revised form): 12th September 2005

Abstract

Progress made in applying agent systems to molecular computational biology is reviewed and strategies by which to exploit agent technology to greater advantage are investigated. Communities of software agents could play an important role in helping genome scientists design reagents for future research. The advent of genome sequencing in cattle and swine increases the complexity of data analysis required to conduct research in livestock genomics. Databases are always expanding and semantic differences among data are common. Agent platforms have been developed to deal with generic issues such as agent communication, life cycle management and advertisement of services (white and yellow pages). This frees computational biologists from the drudgery of having to re-invent the wheel on these common chores, giving them more time to focus on biology and bioinformatics. Agent platforms that comply with the Foundation for Intelligent Physical Agents (FIPA) standards are able to interoperate. In other words, agents developed on different platforms can communicate and cooperate with one another if domain-specific higher-level communication protocol details are agreed upon between different agent developers. Many software agent platforms are peer-to-peer, which means that even if some of the agents and data repositories are temporarily unavailable, a subset of the goals of the system can still be met. Past use of software agents in bioinformatics indicates that an agent approach should prove fruitful. Examination of current problems in bioinformatics indicates that existing agent platforms should be adaptable to novel situations.

INTRODUCTION

Livestock genomics is rapidly transforming into a data-intensive field. Molecular biologists face a daunting task when designing reagents for livestock genomics because of analytical complexity, an ever-expanding database and semantic heterogeneity. Software agents are made for this situation.1 Baylor College of Medicine (BCM) in Houston, Texas has sequenced the bovine genome to approximately six-fold genomic coverage.2,3 The bovine genomics community, the European Bioinformatics Institute (EBI)4 and BCM are currently assembling and annotating the bovine sequence. The Sanger genome sequencing centre will sequence the swine genome over the next couple of years.5 Livestock genomic sequence will have a profound impact on research and technological advancement needed to improve meat production efficiency, quality and safety. The high degree of genomic conservation among mammalian species makes it possible to apply information in one species that was gleaned from another. This comparison of genomes and the mining of information in one species to apply to another is comparative genomics. The proliferation of automated equipment for sequencing, genotyping and studying gene expression indicates that data will be generated at ever increasing rates. The advent of the trace archive,6 Gene Expression Omnibus (GEO)7 and dbSNP8 at the National Center for Biotechnology Information (NCBI) indicates that ever-expanding data resources will be available for mining information from multiple species data on a continuous basis. Data are collected, processed and stored at each laboratory. Much is submitted to...
public repositories. Knowledge in comparative genomics then requires the integration of information from multiple sources. Managing the data, the information and the processes associated with these tasks is difficult, time consuming and often laborious. The use of systems of software agents as part of a comprehensive data management strategy has been explored at several laboratories.9,10

Agents are software entities that have many of the characteristics of their human namesakes. For example, travel agents and real estate agents provide specialised services that make some of our life tasks easier. Agents act on behalf of a client/user, and, it is hoped, are benevolent, and thereby act in their client’s best interests. Software agents are autonomous. They may operate without the direct intervention of a user and they can refuse to act on a request. Agents are reactive; they are able to selectively perceive their environment and respond accordingly. Agents are proactive; they do not simply act in response to a request; they are able to exhibit goal-directed behaviour. Agents are adaptive; they learn and improve with experience.

Another very important characteristic of agents is that they are capable of social behaviour. Agents are able to communicate with users and with other agents. They are able to live and work in ‘communities’. Communication is accomplished with an ontology that defines the semantics and syntax required for meaningful exchanges. Some agent systems are peer-to-peer; they do not rely on communication through a central server as client–server systems do.

Advanced systems may include artificial intelligence capabilities that allow agents to have influential capability. They can act on abstract task specifications using prior knowledge of goals and methodologies. Loosely coupled agent interaction and autonomy make it possible for agents to achieve a subset of their goals in the event that some agents and data repositories are not available.1

Our ability to deal with distributed and heterogeneous information in molecular biology remains primitive. The challenge is to design an information infrastructure capable of accessing global information resources that allows researchers to focus on the content of information rather than its location, its format or different semantics used in different data repositories. Systems of software agents have the potential to build the necessary information infrastructure. Progress in the genomic sciences will be tied to the skill and creativity that we show in establishing an infrastructure to integrate diverse information. Existing distributed systems9,10 for integrating genomics data barely scratch the surface of what is possible. Our aim is to (1) review the progress that has been made in applying agent systems to molecular computational biology and (2) investigate strategies by which we might exploit agent technology to greater advantage.

EXISTING AGENT SYSTEMS

The United States Meat Animal Research Center (MARC) at Clay Center, Nebraska is currently using software agents to assist in the process of submitting sequence data to GenBank. The University of South Carolina (USC) developed the agents on the Java Agent DEvelopment Framework (JADE) platform.9 MARC and USC did the conceptual design jointly. The agents run under Solaris on SUN hardware and interact with Oracle 9i.

The agent system functions as a system of task managers. Task managers query the Oracle database to determine which tasks they are assigned (table TaskAssignments; Figure 1). A subsequent query to the database provides the steps the task manager must execute to complete the task (table TaskCommand). Each step in a task is a command that can be executed from the command prompt. The database tables that support the agent system are shown in Figure 1. A query to the TaskManager table tells the task manager agent how long to sleep after...
completing a task before querying the database for new tasks. The TaskManagerLog table is a log of the work of the task manager. An insert into the table TaskManagerLog is made when each step in started and completed. The column status has a value in (started, succeeded, failed).

The agent architecture is quite general and is database-centric. A task manager can run any sequence of steps (or commands) that can be issued at a command prompt. A task is completely specified by inserting records into three database tables. In essence, an interface to the database also serves as an interface to the agent system. Multiple agents may execute on the same or different host concurrently.

While the USC-MARC system has many desirable properties, an undesirable characteristic is that it has a client–server architecture instead of peer-to-peer. A disadvantage of the client–server architecture relative to peer-to-peer is that the server is a single point of failure. Another disadvantage is that remote agents would need to have access to the database to communicate with the agents in the USC-MARC agent system. With a true peer-to-peer system, remote agents would not need to have access to a central database.

Scientists at MARC are committed to depositing genomic data for livestock in the GenBank repositories at NCBI. The types of data generated by MARC include genomic sequence, expressed sequence tags (EST), single nucleotide polymorphisms (SNP), serial analysis of gene expression (SAGE) and massively parallel signature sequences (MPSS). For the case of EST, the system works as follows:

- The task manager creates a directory in which to store files.
- The task manager is given a file of path names of traces that are to be included in this submission. This file is generated from a database query. Typically it consists of all traces from a library that have not yet been submitted. The database keeps track of dbEST submissions. Traces at MARC are stored in a centre-wide archive on network-attached storage. The task manager runs Phred on traces in the list using the -trim_alt option. Base calls and quality scores are placed in the working directory.
- Next the task manager runs cross_match to identify vector, linkers and adaptors in the base calls.
The task manager then loads sequence with contaminants identified in the previous step masked into a database table.

The task manager runs a Oracle PL/SQL procedure to trim the sequence for quality and remove vector from the ends of trimmed sequence. If the non-vector trimmed sequence contains more than 100 bases, it is written to a submission table and labelled as submittable. Sequences of fewer than 100 bases after trimming and vector removal are written to the table and labelled rejected. Sequences are labelled pending if there exists a vector in the interior of the sequence after trimming for quality.

The task manager then runs another PL/SQL procedure that attempts to 'fix' pending sequences. Occasionally a few bases not labelled as vector may precede vector determined by cross_match. In such cases, these bases are removed if the sequence meets all other requirements.

As currently implemented, this ends the task manager’s work. The EST submission process continues with a manual examination of pending sequences, the running of a perl script to check for low complexity sequence, and a manual examination of the sequences marked as low complexity. The sequences to be submitted to dbEST are then combined with the ancillary information requested by dbEST through a database query and are arranged in the proper format for submission.

An artificial intelligent (AI) planning component could reduce the complexity of launching the USC-MARC agent system and improve its performance. Typically some form of AI planning activity governs task allocation across multiple agents and planning agents use domain model knowledge in their planning activity. AI planning is choosing and ordering the sequence of actions (or steps) needed to achieve objectives. AI planning has been used to develop plans for hospitals, chemical plants, space exploration and logistics support in the conduct of war. An AI planning component would make it possible to launch a task by specifying what is to be done without having to identify and order the steps required to do it. This would make the system more relevant and available to users who know what they want done but do not know how to do it.

myGrid16 is bioinformatics middleware that is able to plan and execute workflows based on specification of objectives and not procedures. The domain knowledge required for planning activities could be obtained from an ontology agent with access to web services such as BioMoby. The web services integration gateway (WSIG) of JADE1 could provide this access.

Agent systems that access distributed data resources require an ontology that has been accepted by all components of the system. Working groups composed of parties with vested interests in the outcome make most attempts at global domain-specific ontologies. An example of one of these working groups was the Model Organism Bring Your own Database Interface Conference (MOBY-DIC) held September 2001. Agent systems that operate locally, such as the USC-MARC system, essentially use the entity-relationship (ER) model of the database as the ontology. An ontology for two heterogeneous relational databases may be formed by mapping the ER model for one database onto the ER model for the other if sufficient metadata exist to build the mapping. This process becomes impractical as the number of resources increases, hence the need for a generally accepted ontology. A similar approach is the ‘wrapping’ or translation of distinct resources to a common ontology.

Advances in artificial intelligence may make it possible to address complex problems through executing a sequence of sub-tasks. Sub-tasks can be thought of...
as in silico experiments. It may be impossible to know what the sequence of sub-tasks should be a priori. Hope is on the way. King et al.\textsuperscript{18} were able to use intelligent software, Robot Scientist, to direct a robot in conducting experiments to determine the aromatic amino acid biosynthesis pathway in baker’s yeast (\textit{Saccharomyces cerevisiae}). Robot Scientist performed at least as well as the best postdoctoral scientists and graduate students studied. Robot Scientist includes routines for hypothesis generation, experiment selection, experiment planning and experiment interpretation. Whelan and King\textsuperscript{19} concluded that it is indeed possible for conclusions or hypotheses to be derived automatically from raw experimental results using intelligent software. If this works in the laboratory, it should certainly work for in silico experiments. This raises exciting possibilities for automated knowledge generation given the large volume of raw genomics data currently making its way into public databases.

Decker et al.\textsuperscript{10} developed a multi-agent system called BioMAS to automate annotation, EST processing and metabolic pathway reasoning. The system eliminates tedious manual analyses and makes the annotations available to other researchers or agent systems. BioMAS uses a generic agent platform called DECAF (Distributed, Environment-Centered Agent Framework),\textsuperscript{20} which consists of RETSINA\textsuperscript{21} and TAEMS.\textsuperscript{22} RETSINA consists of three classes of agents – information extraction agents, task agents (apparently, these agents drive processing steps similar to the USC-MARC system) and interface agents to deal with the end user. The Decker and Schmidt group has subsequently developed public websites for chicken (GallusKB), herpes viruses (HVDB) and fungi (FungiKB) based on principles learned while developing BioMAS.\textsuperscript{23} In addition, they developed a tool (GoFigure) to assist in annotating novel sequences with Gene Ontology\textsuperscript{24} terms. These websites utilise distributed queries but they do not rely directly on software agents. In a sense, they are light versions of BioMAS. This was done because response time was more important for these specialised web resources than flexibility. One of the strengths of BioMAS is query flexibility. For example, BioMAS has a component that permits one to identify EST contigs that are likely involved in a given metabolic pathway (eg pentose phosphate pathway) using KEGG.\textsuperscript{25} It is also possible to initiate queries with an anonymous bit of sequence, a gene symbol or a Gene Ontology term.

**A NEW ROLE FOR AGENTS**

An important role for bioinformatics is helping biologists design reagents for further research. The process of designing reagents has increased in complexity with the advent of whole genomic sequencing in cattle and swine. A molecular biologist in our laboratory recently remarked that it takes longer to design primers for his experiments in swine genetics than it did a year or two ago. He was basically implying that there is now too much data to look at. In the old days, he could design a plate of primers from EST and be confident that the information that he generated would be novel and unique. Nowadays he is not sure if he will be generating new information even after looking at reams of data. It is becoming more and more difficult for biologists to efficiently utilise all of the available information relevant to a particular genetics problem.

We now discuss a genomics problem that was recently presented to us and suggest how the JADE\textsuperscript{1} platform might be configured to address it. It is not our intention to endorse the JADE platform over DECAF\textsuperscript{20} or others. We choose to discuss JADE because we are more familiar with it. A scientist at MARC is interested in studying Uncoupling Protein 1 (UCP1) in cattle because of its role in brown fat metabolism and the possible role that it may play in the survival of young calves during cold weather.\textsuperscript{26} He was wondering if we could find the entire
It is difficult to find all of the relevant data pertaining to a chromosomal region. It is difficult to find all of the relevant data. Indeed, there is no guarantee that we have found all of the data. We will now address three questions. First, exactly what were we able to find out about bovine UCP1 and how might this information be used? Second, what process was used to gather the data? Third, can software agents facilitate data gathering?

For the cow, we were able to identify six putative exons (Figure 2) even though MapViewer only shows five (exons 2 to 6). This is because the latest bovine genome assembly had not been processed through NCBI's pipeline. Three key pieces of data support the more complete bovine UCP1 gene structure shown in Figure 2. First, a genomic sequence (AJ275981; submitted in 1999) joins the 2005 vintage Baylor sequence contigs – AAFC02223393 (containing exon 1) and AAFC02134563 (containing exons 2 to 6). Second, a partial expressed sequence (X14064; published in 1989) includes part of exon 1 and all of exons 2 to 6. Third, an open reading frame on AAFC02223393 has 74 per cent amino acid similarity with exon 1 for human based on tBlastx. In addition, this open reading frame matches the predicted amino acid sequence of X14064 in the published reading frame.

Molecular biologists in our laboratory would like to have at least 5 kb of sequence flanking both ends of UCP1 to look for promoters and cis regulatory elements. We have identified 5–6 kb of flanking sequence 3' to bovine UCP1 but only 500 bp of contiguous sequence on the 5' end. We do not know where the 5' untranslated region begins because we have not found expressed sequence for the 5' end of the transcript. The availability of the bovine bacterial artificial chromosome (BAC) physical map including end sequences may now make it possible to identify a bovine clone containing UCP1 from which to target the sequencing efforts. We identified five clones overlapping UCP1 by comparing BAC end sequence to Baylor contig sequences (using MegaBlast) and Baylor contig sequence to the human contigs (Figure 3). Any one of these clones can be used to target sequencing of UCP1 flanking regions. Strategies include direct sequencing of the BAC clone, sub-cloning and sequencing of the sub-clones, or PCR amplification followed by sequencing. Baylor sequencing contigs in the 5' flanking region of UCP1 are AAFC02084557 and AAFC02223393. Based on alignment with the human contig (NT_016354), we predict that the distance between adjacent ends of AAFC02084557 and AAFC02223393 is about 2 kb. Unless this region is a lot

**Figure 2:** Putative gene structure for bovine Uncoupling Protein 1 (UCP1) with associated evidence. Dots above line segments representing AJ275981 and AF139921 are SNP positions annotated on the GenBank records.
longer in bovine than in human contigs, it should be possible to amplify across the unsequenced region using polymerase chain reaction (PCR).

The information presented in Figures 2 and 3 was obtained using queries depicted in Table 1. We identified the segment of the human genomic contig containing UCP1 using MapViewer. We then identified bovine sequence contigs that were similar to it using MegaBlast. BAC end sequences were linked to bovine contigs, which in turn were linked to human contigs using MegaBlast. The advantage of this two-step procedure is it makes it possible to consider only high-stringency sequence matches. Hence, a high percentage of the BAC ends can be unequivocally mapped to the human genome. Once a BAC end was matched to a bovine contig, the other end was compared with the bovine contigs to attempt to create a physical link between the bovine contigs as well as order them (Figure 3).

It should be feasible to develop JADE agents that call routines from Bioperl and/or workflows from Taverna to accomplish the above tasks. The necessary infrastructure is available in the public domain. Clearly, we need an agent or agents capable of submitting MegaBlast jobs and retrieving the results. We need an agent capable of assembling sequences (e.g., it might use Phrap) such as what was done to create the merged sequence represented on the horizontal axis of Figure 2. Agents capable of displaying the results would be necessary. It would be desirable for agents to be able to choose among different sites from which to run MegaBlast so as not to overload busy sites. User agents would be needed to communicate the user’s wishes to the agent community. It would be desirable for the system to be distributed on multiple sites with some redundant agents so that there is not a single point of failure. Agents would need to communicate what they are doing to the

Table 1: Types of query sequence and databases for MegaBlast

<table>
<thead>
<tr>
<th>Query sequence</th>
<th>Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segment of human genomic reference sequence</td>
<td>Bos taurus contig sequences</td>
</tr>
<tr>
<td>Bos taurus contig sequence</td>
<td>Human genomic reference sequences</td>
</tr>
<tr>
<td>Bos taurus BAC ends</td>
<td>Non-redundant database (NR)</td>
</tr>
<tr>
<td>Bos taurus BAC end</td>
<td>dbEST</td>
</tr>
<tr>
<td></td>
<td>Bos taurus contig sequences</td>
</tr>
</tbody>
</table>
Agents need to be programmed to respond to communications appropriately so that two or more agents do not tackle the same job. An analysis agent is needed that looks at the data returned by the other agents, evaluates progress towards the goal, and determines whether additional queries are needed. It may be preferable to divide this work among several agents, perhaps using an AI planning approach.\textsuperscript{14}

WHY ARE SOFTWARE AGENTS NOT IN WIDESPREAD USE?

Clearly, the previous efforts in applying software agents to bioinformatics demonstrate the potential of these systems. Just as clearly, use of software agents in bioinformatics has not caught on. For software agents to shine, there needs to be a distributed genomics community using them so that people are able to leverage one another’s efforts. Sending agents out over the network looking for information is fruitless if there are not other agents out there with relevant data for them to talk to.

The under-utilisation of agents in bioinformatics may be due to the availability of alternatives such as Bioperl. If this is an issue for people, it is more psychological than technical. Software agents and Bioperl are not mutually exclusive; for example, some USC-MARC agents call Bioperl routines. Likewise, tools such as Biodas\textsuperscript{35} and Taverna are more complements to software agents than competitors.

One of the most frequently quoted problems hindering semantic integration of distributed resources is the lack of good, commonly agreed upon ontologies. Communication among software agents depends on the availability of a shared ontology. Ontologies can be accessed from web services if they exist to begin with. Clearly, a community effort is required to develop and maintain ontologies. The BioMoby\textsuperscript{17} project is definitely a step in the right direction.

Another concern is that processor speed and network bandwidth are not currently sufficient to support communities of software agents. Clearly, computer resource constraints should influence the behaviour of agents but they should not be showstoppers. For example, you would not want agents doing whole chromosome sequence assemblies on undersized computers on the fly while interactive users are trying to access the same system. The reasoning capability of agents allows them to deal with constraints such as limitations in processor speed and network bandwidth.\textsuperscript{1}

Another concern is internet security. What if someone creates an agent (or virus) that dupes our agents into acting maliciously? Clearly, there are risks. Platforms such as JADE have security components to help deal with this problem. This may be an area where software agents have the upper hand over other types of middleware. If software agents can be made to be ‘internet-wise’ by their developers then they may be less prone to security problems than other types of middleware.

One commonly held point of view is that software agents constitute interesting research in computer science but the platforms and tools are simply not ready for real world applications such as bioinformatics. It is thought that people working on serious bioinformatics problems simply do not have the time or patience to deal with software as glitchy and experimental as agent platforms. A recent white paper\textsuperscript{1} written by JADE developers disputes this idea. They claim that FIPA-compliant Foundation for Intelligent Physical Agents\textsuperscript{36}) agent platforms such as JADE are currently moving from software research towards advanced business applications.

Technology transfer is a people-centred process, which depends as much on organisation, training and getting the word out as it does on the readiness of the technology for application. We suspect that the lack of software agent implementations in bioinformatics is caused by an insufficient critical mass of
computational biologists that know enough about software agents to consider incorporating them in their work. In addition, those of us using software agents probably have not done an adequate job of getting the word out.

References


2. Human Genome Sequencing Project at Baylor College of Medicine, Bovine Genome Project (URL: http://www.hgsc.bcm.tmc.edu/projects/bovine/ [accessed 21st July, 2005]).


4. Wellcome Trust Sanger Institute Ensembl Genome Browser for the cow (URL: http://pre.ensembl.org/Bos_taurus/ [accessed 21st July, 2005]).


31. Michael Smith Genome Sciences Centre, A BAC fingerprint map of the bovine genome (URL: http://www.bcgsc.ca/lab/mapping/bovine [accessed 20th July, 2005]).


33. Bioperl (URL: http://www.bioperl.org/ [accessed 22nd July, 2005]).

34. Taverna (URL: http://taverna.sourceforge.net/ [accessed 21st July, 2005]).

35. Biodas (URL: http://biodas.org/ [cited accessed 22nd July, 2005]).